FIGURE 1

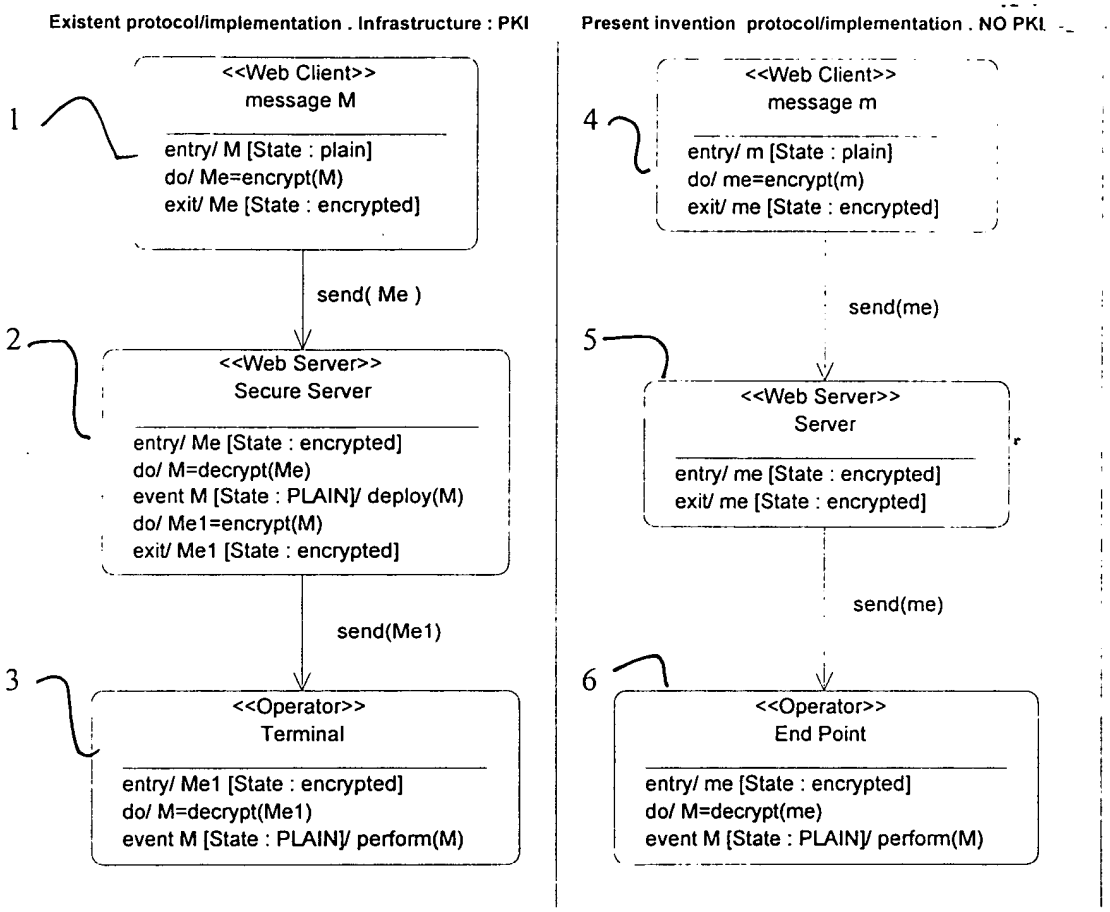
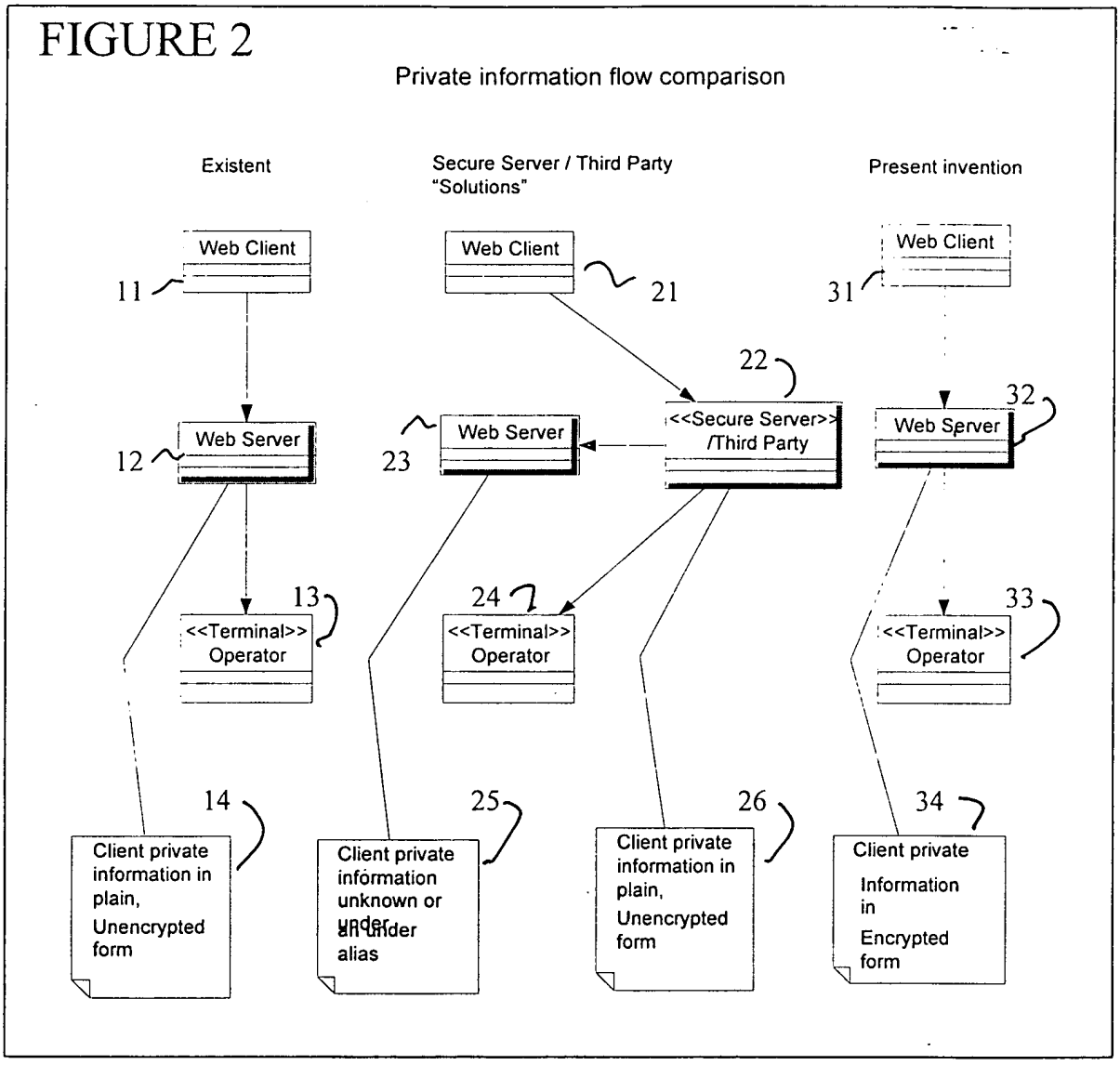

Existent protocol/implementation . Infrastructure : PKI
Present invention protocol/implementation . NO PKI
1
<<Web Client>>
message M
entry/ M [State : plain]
do/ Me=encrypt(M)
exit/ Me [State : encrypted]
send(Me)
2
<<Web Server>>
Secure Server
entry/ Me [State : encrypted]
do/ M=decrypt(Me)
event M [State : PLAIN]/ deploy(M)
do/ Me1=encrypt(M)
exit/ Me1 [State : encrypted]
send(Me1)
3
<<Operator>>
Terminal
entry/ Me1 [State : encrypted]
do/ M=decrypt(Me1)
event M [State : PLAIN]/ perform(M)
4
<<Web Client>>
message m
entry/ m [State : plain]
do/ me=encrypt(m)
exit/ me [State : encrypted]
send(me)
5
<<Web Server>>
Server
entry/ me [State : encrypted]
exit/ me [State : encrypted]
send(me)
6
<<Operator>>
End Point
entry/ me [State : encrypted]
do/ M=decrypt(me)
event M [State : PLAIN]/ perform(M)

FIGURE 2


Private information flow comparison
Existent
Secure Server / Third Party "Solutions"
Present invention
Web Client
11
Web Client
21
Web Client
31
22
Web Server
12
Web Server
23
<<Secure Server>> /Third Party
Web Server
32
13
<<Terminal>> Operator
24
<<Terminal>> Operator
33
<<Terminal>> Operator
14
Client private information in plain, Unencrypted form
25
Client private information unknown or under an alias
26
Client private information in plain, Unencrypted form
34
Client private Information in Encrypted form

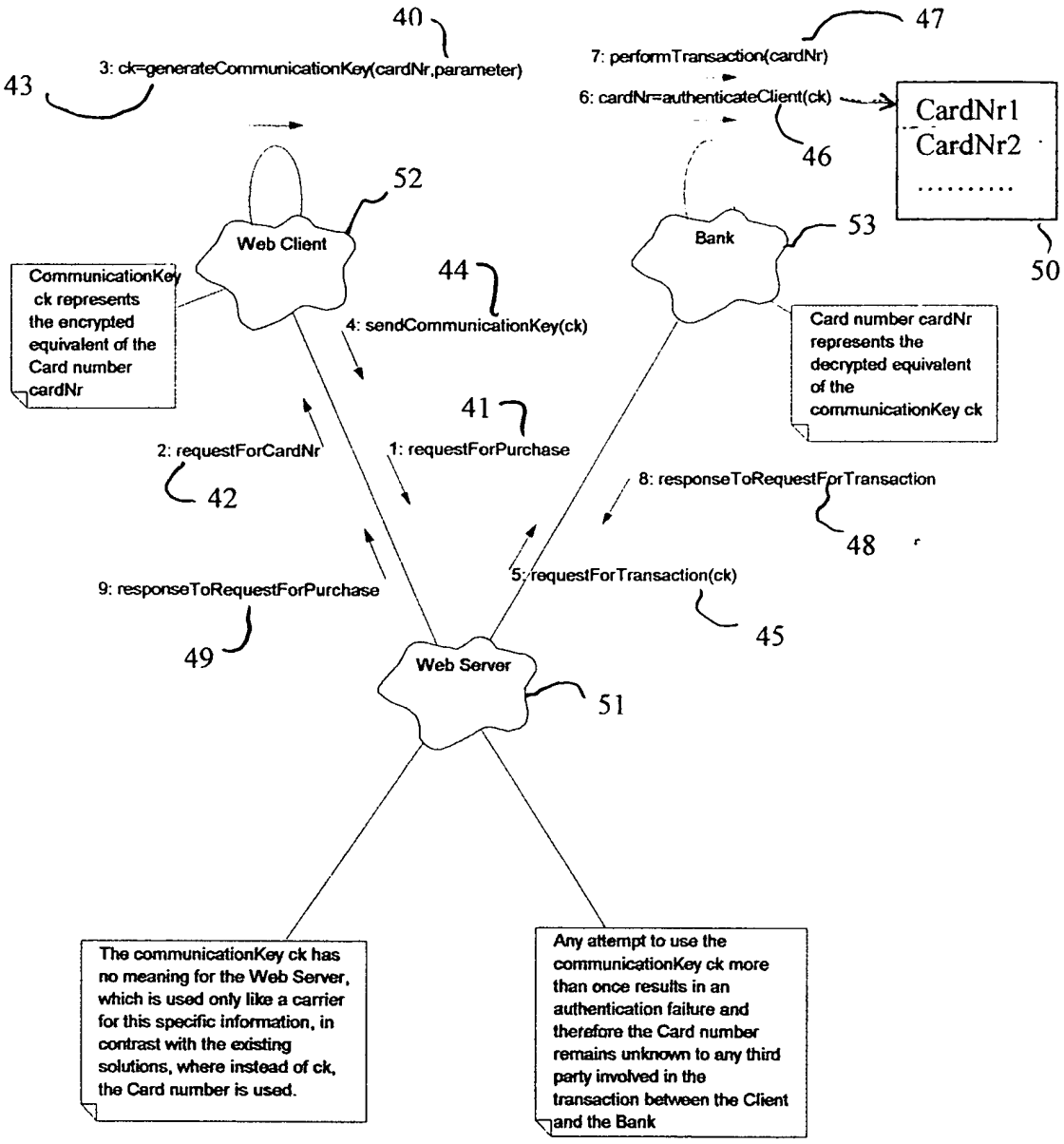

40
3: ck=generateCommunicationKey(cardNr,parameter)
43
7: performTransaction(cardNr)
47
6: cardNr=authenticateClient(ck)
46
CardNr1
CardNr2
.....
50
52
Web Client
53
Bank
CommunicationKey ck represents the encrypted equivalent of the Card number cardNr
44
4: sendCommunicationKey(ck)
Card number cardNr represents the decrypted equivalent of the communicationKey ck
41
2: requestForCardNr
42
1: requestForPurchase
8: responseToRequestForTransaction
48
9: responseToRequestForPurchase
49
5: requestForTransaction(ck)
45
Web Server
51
The communicationKey ck has no meaning for the Web Server, which is used only like a carrier for this specific information, in contrast with the existing solutions, where instead of ck, the Card number is used.
Any attempt to use the communicationKey ck more than once results in an authentication failure and therefore the Card number remains unknown to any third party involved in the transaction between the Client and the Bank

FIGURE 3

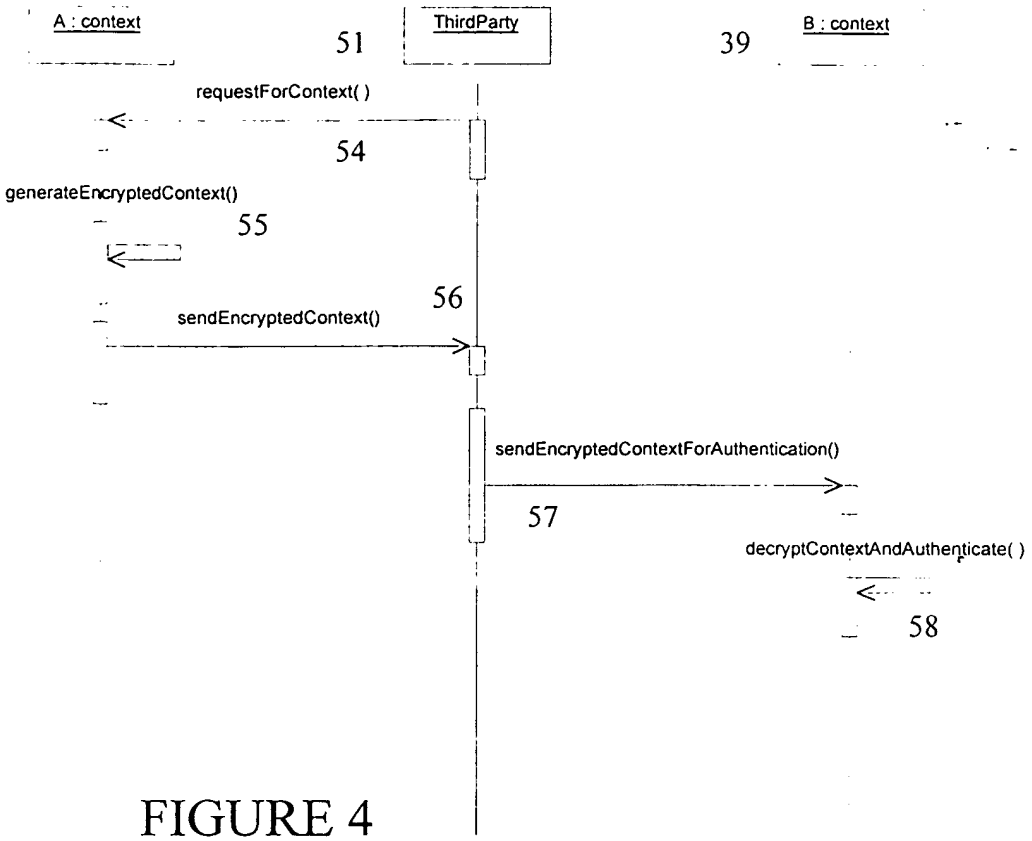

38
A : context
51
ThirdParty
39
B : context
requestForContext( )
54
generateEncryptedContext()
55
56
sendEncryptedContext()
sendEncryptedContextForAuthentication()
57
decryptContextAndAuthenticate( )
58

FIGURE 4

FIGURE 5

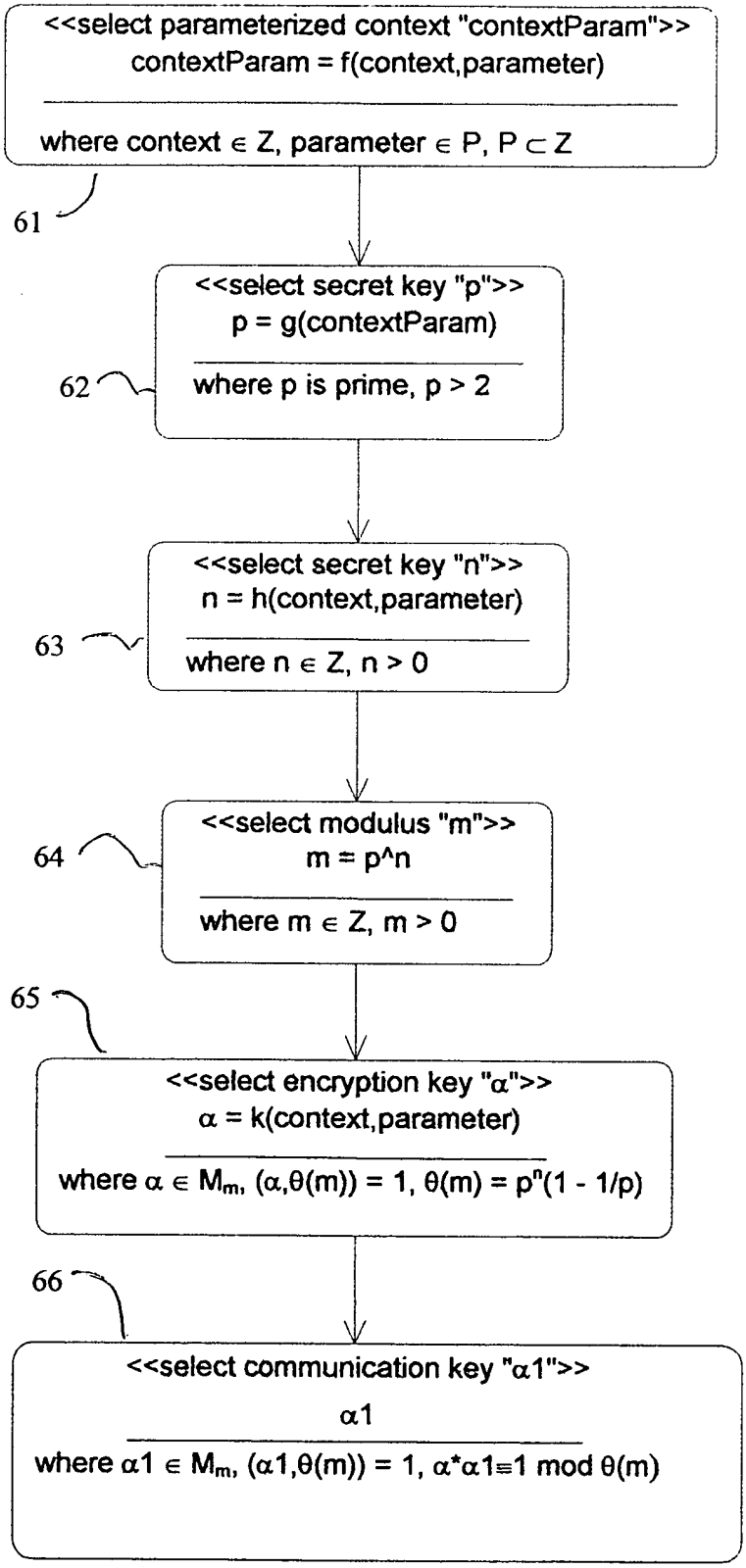

<<select parameterized context "contextParam">>
contextParam = f(context,parameter)
where context ∈ Z, parameter ∈ P, P ⊂ Z
61
<<select secret key "p">>
p = g(contextParam)
where p is prime, p > 2
62
<<select secret key "n">>
n = h(context,parameter)
where n ∈ Z, n > 0
63
<<select modulus "m">>
m = p^n
where m ∈ Z, m > 0
64
65
<<select encryption key "α">>
α = k(context,parameter)
where α ∈ Mm, (α,θ(m)) = 1, θ(m) = p^(1 - 1/p)
66
<<select communication key "α1">>
α1
where α1 ∈ Mm, (α1,θ(m)) = 1, α*α1≡1 mod θ(m)

FIGURE 6

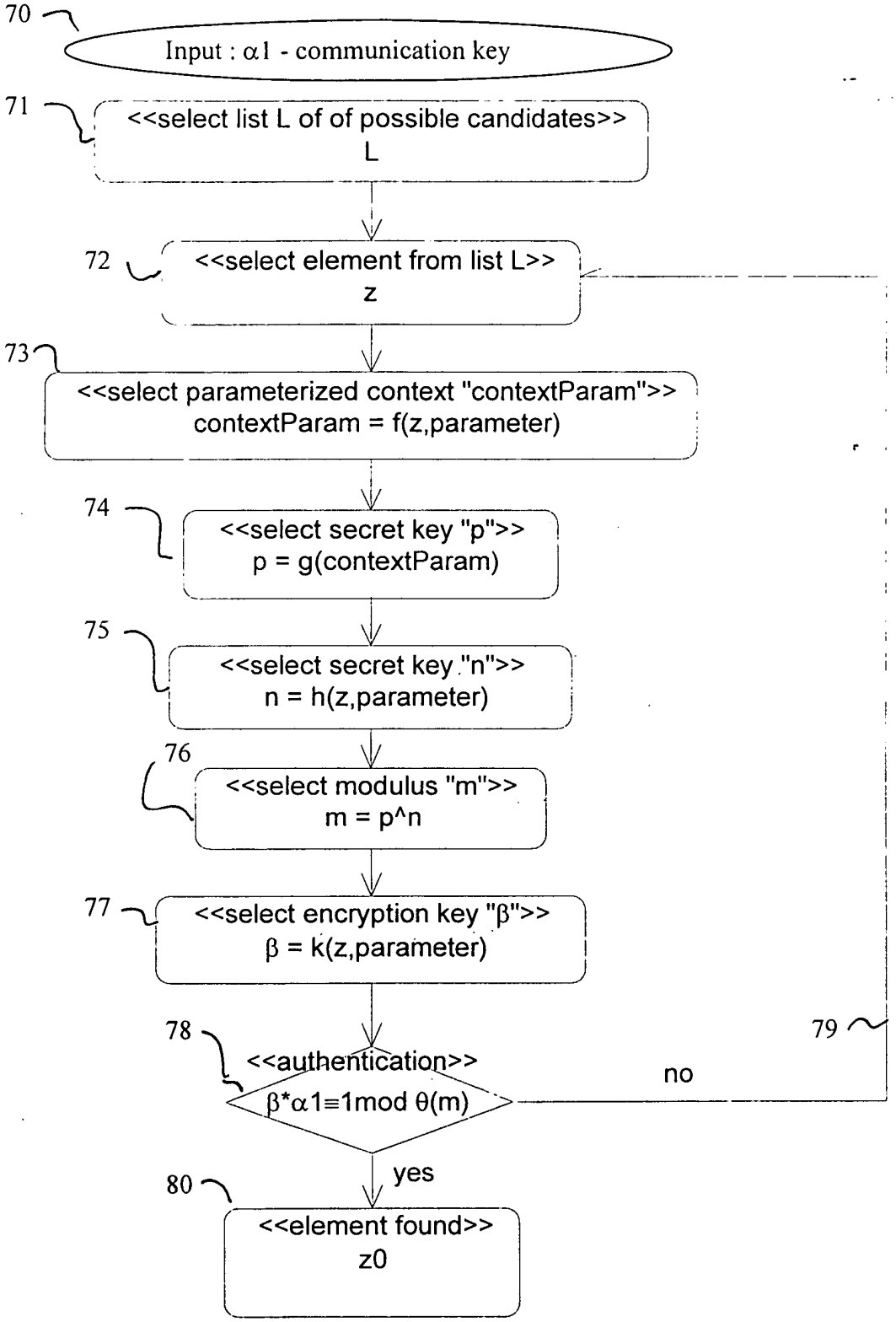
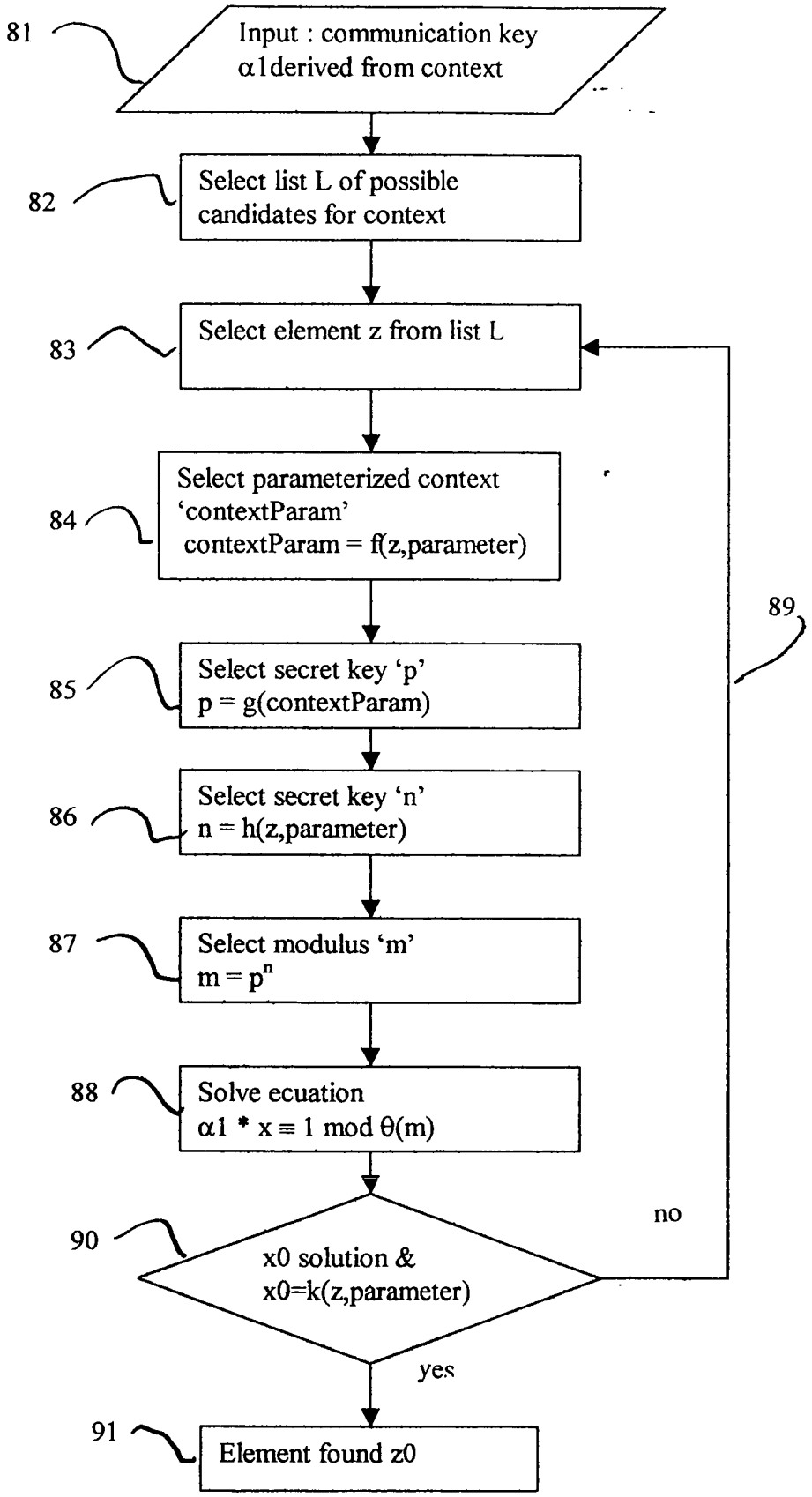

70
Input : α1 - communication key
71
<<select list L of of possible candidates>>
L
72
<<select element from list L>>
z
73
<<select parameterized context "contextParam">>
contextParam = f(z,parameter)
74
<<select secret key "p">>
p = g(contextParam)
75
<<select secret key "n">>
n = h(z,parameter)
76
<<select modulus "m">>
m = p^n
77
<<select encryption key "β">>
β = k(z,parameter)
78
<<authentication>>
β*α1≡1mod θ(m)
no
79
yes
80
<<element found>>
z0

FIGURE 7


81
Input : communication key
α1derived from context
82
Select list L of possible
candidates for context
83
Select element z from list L
84
Select parameterized context
'contextParam'
contextParam = f(z,parameter)
85
Select secret key 'p'
p = g(contextParam)
86
Select secret key 'n'
n = h(z,parameter)
87
Select modulus 'm'
m = p^n
88
Solve ecuation
α1 * x ≡ 1 mod θ(m)
90
x0 solution &
x0=k(z,parameter)
no
89
yes
91
Element found z0

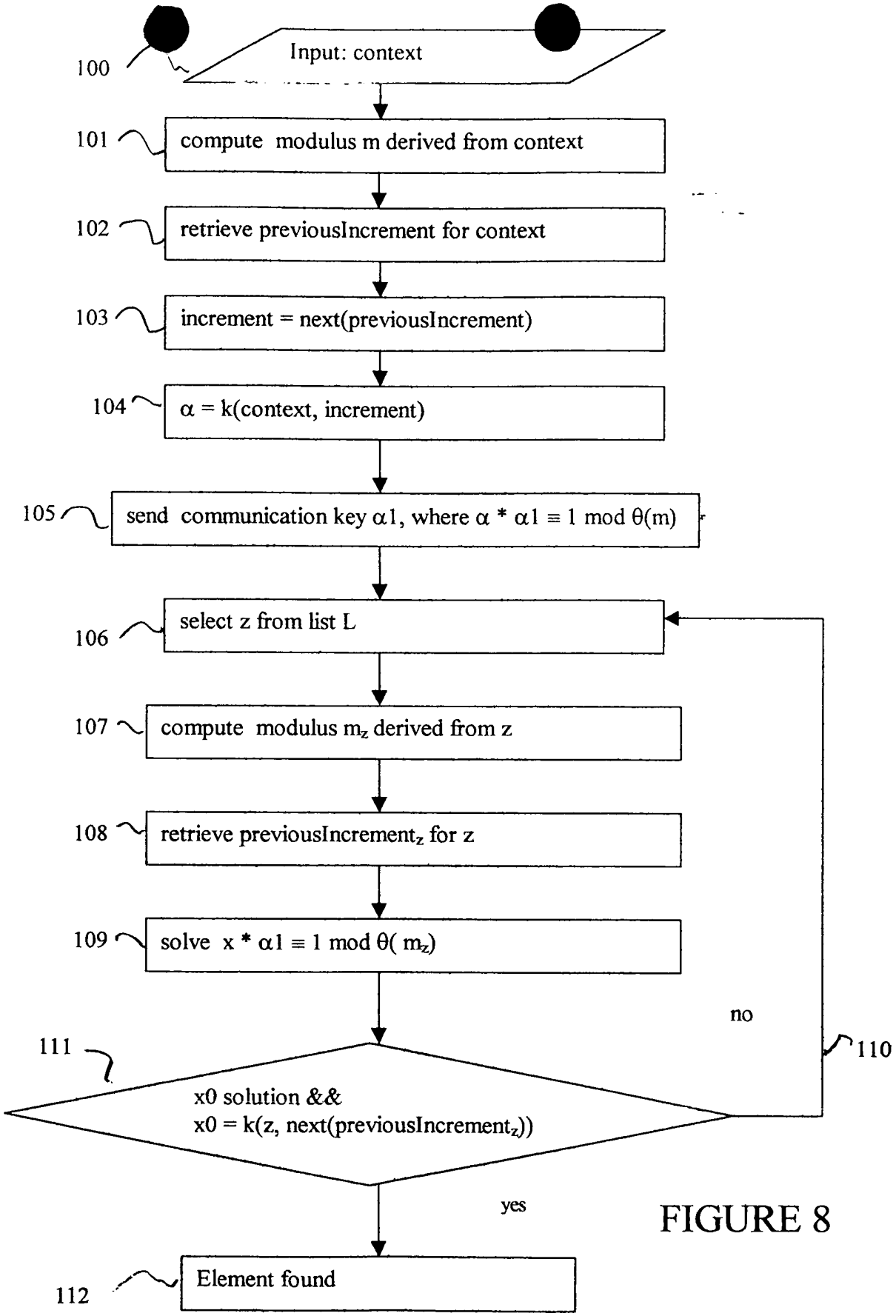

100
Input: context
101
compute modulus m derived from context
102
retrieve previousIncrement for context
103
increment = next(previousIncrement)
104
α = k(context, increment)
105
send communication key α1, where α * α1 ≡ 1 mod θ(m)
106
select z from list L
107
compute modulus m_z derived from z
108
retrieve previousIncrement$_z$ for z
109
solve x * α1 ≡ 1 mod θ(m_z)
111
x0 solution &&
x0 = k(z, next(previousIncrement$_z$))
no
110
yes
112
Element found

FIGURE 8

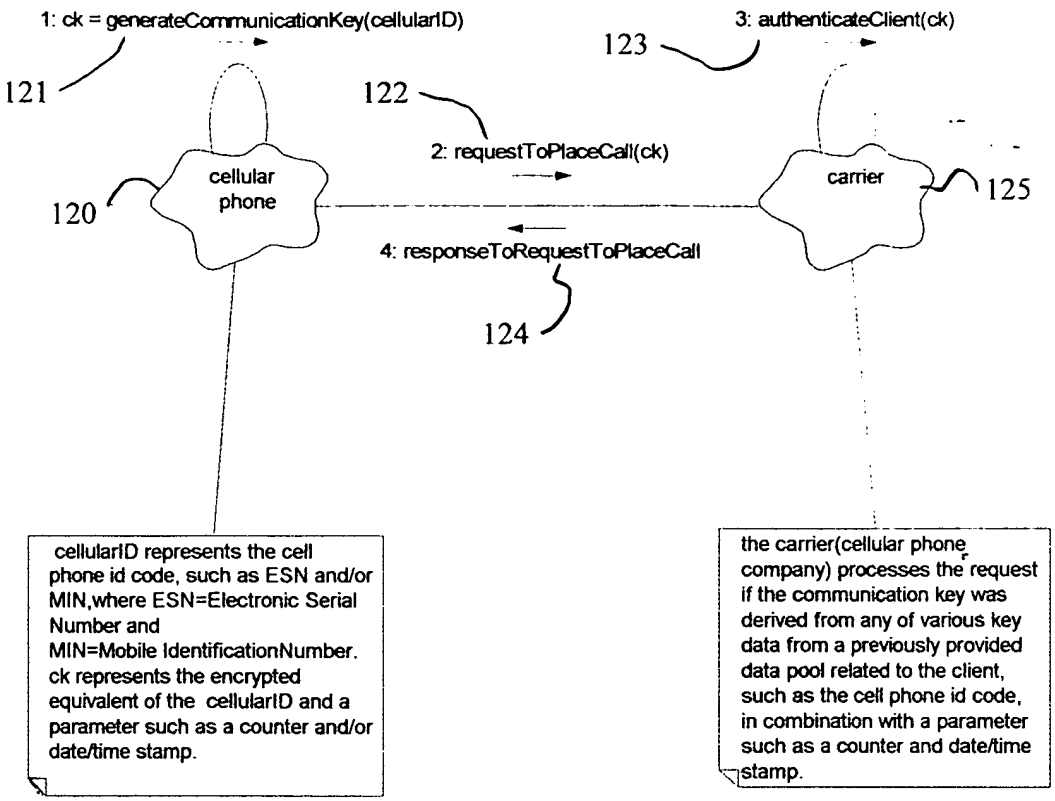

1: ck = generateCommunicationKey(cellularID)
121
3: authenticateClient(ck)
123
122
2: requestToPlaceCall(ck)
cellular phone
120
carrier
125
4: responseToRequestToPlaceCall
124
cellularID represents the cell phone id code, such as ESN and/or MIN,where ESN=Electronic Serial Number and MIN=Mobile IdentificationNumber. ck represents the encrypted equivalent of the cellularID and a parameter such as a counter and/or date/time stamp.
the carrier(cellular phone company) processes the request if the communication key was derived from any of various key data from a previously provided data pool related to the client, such as the cell phone id code, in combination with a parameter such as a counter and date/time stamp.

FIGURE 9